REMARKS

Applicant submits this Response to the Office Action mailed August 14, 2003. Applicant has amended claims 1, 8, 10 and 14. Claims 1-34 remain pending. No new matter has been added.

In paragraph 3 of the Office Action, the Examiner has objected to claims 1 and 14 because of informalities. Applicant has amended claims 1 and 14 to correct the errors identified by the Examiner, and respectfully requests that the objection be withdrawn.

In paragraph 4 of the Office Action, the Examiner has objected to claim 10 as being of improper dependent form. Specifically, the Examiner has noted that "If the jump instruction of claim 10 refers to a different jump instruction that the jump instruction of claim 8, such a distinction is unclear as the claims are presented, and a different identifier is required to make the distinction clearer." (Office Action, p. 3.) Applicant has amended claims 8, 10 and 14 to use different identifiers for each jump instruction, and believes these amendments overcome the Examiner's objections. Applicant respectfully requests that the Examiner withdraw the objection.

In paragraphs 5-6 of the Office Action, the Examiner has rejected claims 1-3, 8-10, 15-19 and 21-23 under 35 U.S.C. § 102(b) as being anticipated by U.S. Patent No. 5,359,721 to Kempf et al. ("the Kempf reference"). In paragraph 7 of the Office Action, the Examiner has rejected claims 28-29 and 32-34 under U.S.C. § 102(e) as being anticipated by U.S. Patent No. 6,546,546 to Van Doorn ("the Van Doorn reference"). In paragraphs 8-9 of the Office Action, the Examiner has rejected claims 4-7, 11, 20, 24-27 and 30-31 under U.S.C. § 103(a) as being unpatentable over the Kempf reference in view of the Van Doorn reference. In paragraph 10 of the Office Action, the Examiner has rejected claims 12-14 under 35 U.S.C. § 103(a) as being unpatentable over Kempf in view of Van Doorn and in view of U.S. Patent No. 6,199,152 to Kelly et al. ("the Kelly reference"). Applicant respectfully requests that the Examiner reconsider these rejections based on the following.

The Kempf reference describes "[a] method and apparatus for a process executing in non-supervisor mode to perform cross address space dynamic linking." (Kempf, col. 4, lines 65-67.) Address space is managed through an "address space manager," which makes "address space

objects" available to client processes. (Id., col. 6, lines 54-56.) The Kempf reference states that the client process initially obtains the desired address space and a code table for the address space. (Id., col. 8, lines 60-66.) If the code table indicates that the desired program code segment has not been linked in the address space, the client process obtains the program code segment object, builds a symbol address table for the address space, maps the program code segment into the address space and into its own address space and links the program code segment. (Id., col. 9, lines 3-17.) The symbol address table is described as "comprising name to address cross reference entries for the address space." (Id., col. 7, lines 10-16.) Once linked, the client process locates the initialization function of the program code segment, starts a new execution thread and transfers control to the initialization function. (Id., col. 9, lines 18-25.) The address space object comprises an address space and an "interface for performing limited operations on the address space on behalf of client processes operating in non-supervisor mode." (Id., col. 6, lines 60-64.)

The Van Doorn reference describes a JVM "which uses hardware protection domains to transparently and securely protect Java applets." (Van Doorn, col. 2, lines 43-44.) The "protection domains" are described in the Van Doorn reference as "a mapping of virtual to physical pages together with a set of domain specific events." (Id., col. 4, lines 64-65.) Each protection domain has "a view of its own subtree of the name space." (Id., col. 5, lines 21-23.) Also, the Van Doorn reference notes that the "view includes a set of physical memory pages to virtual mappings." (Id., col. 8, lines 47-49.) In order to access methods not within a caller's protection domain, a cross protection domain call must be performed, which must "pass through the Java Nucleus to control access and used [sic] CPU resources." (Id., col. 8, lines 20-22.)

The Kelly reference describes a microprocessor having a "morph host" hardware portion and an emulating software portion, such that various microprocessor architectures can be emulated. (Kelly, col. 11, lines 6-15.) One of the features of the microprocessor described in the Kelly reference is an exception handling process that emulates exception handling on the target microprocessor. (Id., col. 13, lines 8-61.) The Kelly reference states that exceptions generated by the target software are to be handled by emulating the target processor hardware facilities using the code morphing software. (Id., col. 13, lines 42-48.)

In contrast to the Kempf, Van Doorn and Kelly references, the present invention as recited in claim 1 is a method that includes

generating a link stub for the symbol reference when the symbol reference is to an external location to access the external location; and

redirecting the instruction to the link stub.

The Kempf, Van Doorn and Kelly references neither teach nor suggest (either alone or in combination) such a method. The Kempf reference provides no discussion of generating a link stub, or of redirecting an instruction to a link stub. At best, the Kempf reference describes the well known process of directly linking program code and data though the use of a symbol table—"The generated symbol address table is in turn used to facilitate linking a program code segment to another program code segment or process." (Kempf, col. 7, lines 16-18.) Furthermore, the Kempf reference merely describes transferring control to the linked program at its starting address (or initialization function), (id., col. 8, lines 53-56; col. 9, lines 21-25), not redirecting an instruction to a link stub. The Van Doorn and Kelly references also provide no discussion of generating a link stub, or of redirecting an instruction to the link stub.

In further contrast to the Kempf, Van Doorn and Kelly references, the present invention as recited in claim 8 is a method that includes

executing a first jump instruction in the number of instructions that refers to a link stub corresponding to an external location in a second domain;

executing the link stub.

The Kempf, Van Doorn and Kelly references neither teach nor suggest such a method. The Kempf reference provides no discussion of executing a jump instruction that refers to a link stub corresponding to an external location in a second domain. At best, the Kempf reference describes the well known process of directly linking program code and data through the use of a symbol table, not executing a jump instruction that refers to a link stub. Nor does the Kempf reference describe executing a link stub, as no discussion of link stubs is present in the Kempf reference. The Van Doorn and Kelly references also do not describe executing a jump instruction that refers to a link stub corresponding to an external location in a second domain, and executing the link stub.

In further contrast to the Kempf, Van Doorn and Kelly references, the present invention as recited in claim 15 is a computer system comprising:

> a system space having a number of memory locations;
>
> a number of protection domains, at least one of the number of protection domains owning a portion of the system space.

The Kempf, Van Doorn and Kelly references neither teach nor suggest such a method. The Kempf reference provides no discussion of "protection domains," much less at least one protection domain owning a portion of the system space. At best, the Kempf reference describes address space objects that allow execution in non-supervisor mode. The Van Doorn reference discusses protection domains having a view of an address space, however, the Van Doorn reference does not discuss at least one protection domain owning a portion of the system space. The Kelly reference provides no description of protection domains.

With respect to claim 28, the present invention as recited in claim 28 is a protection domain comprising:

a memory space; and

a protection view designating a set of protection domains for unrestricted memory access.

The Van Doorn reference neither teaches nor suggests such a protection domain. Nowhere does Van Doorn describe a protection domain having a protection view that designates a set of protection domains for unrestricted memory access. At best, the Van Doorn reference describes protection domains each having a protection view of "a set of physical memory pages to virtual mappings"(id., col. 8, line 48), but not a protection view designating a set of protection domains for unrestricted memory access. In fact, the Van Doorn reference describes the complete opposite of the protection domain of claim 28—the Van Doorn reference notes that cross protection domain method access must pass through the Java Nucleus, which restricts memory access. Moreover, the Kempf and Kelly references provide no discussion of protection domains or protection views.

Based on the foregoing, Applicant believes that the Kempf, Van Doorn and Kelly references (either individually or in combination) do not teach or suggest all of the elements of

claims 1, 8, 15 and 28, and therefore Applicant believes claims 1, 8, 15 and 28 to be patentable over the Kempf, Von Doorn and Kelly references. Applicant respectfully requests that the Examiner withdraw the rejections of these claims. As claims 2-7 depend from claim 1 (and therefore include all of the limitations of claim 1), claims 9-14 depend from claim 8 (and therefore include all of the limitations of claim 8), claims 16-27 depend from claim 15 (and therefore include all of the limitations of claim 15), and claims 29-34 depend from claim 28 (and therefore include all of the elements of claim 28), Applicant believes claims 2-7, 9-14, 16-27 and 29-34 to be patentable over the Kempf, Van Doorn and Kelly references as well, and respectfully requests that the rejections of these claims be withdrawn as well.

In light of the foregoing, claims 1-34 are believed to be in condition for allowance. All issues raised by the Examiner having been addressed, a early and favorable action on the merits is earnestly solicited. Should the Examiner desire further discussion of Applicant's remarks, Applicant (via the undersigned) is available for telephonic interview at the Examiner's convenience.

Respectfully submitted,

Dated: December 15, 2003

Joseph R. Palmieri, Esq. (Reg. No. 40,760)
136 Turtle Cove Lane
Huntington, NY 11743
(631) 678-2306

Attorney for Applicant
WIND RIVER SYSTEMS, INC.